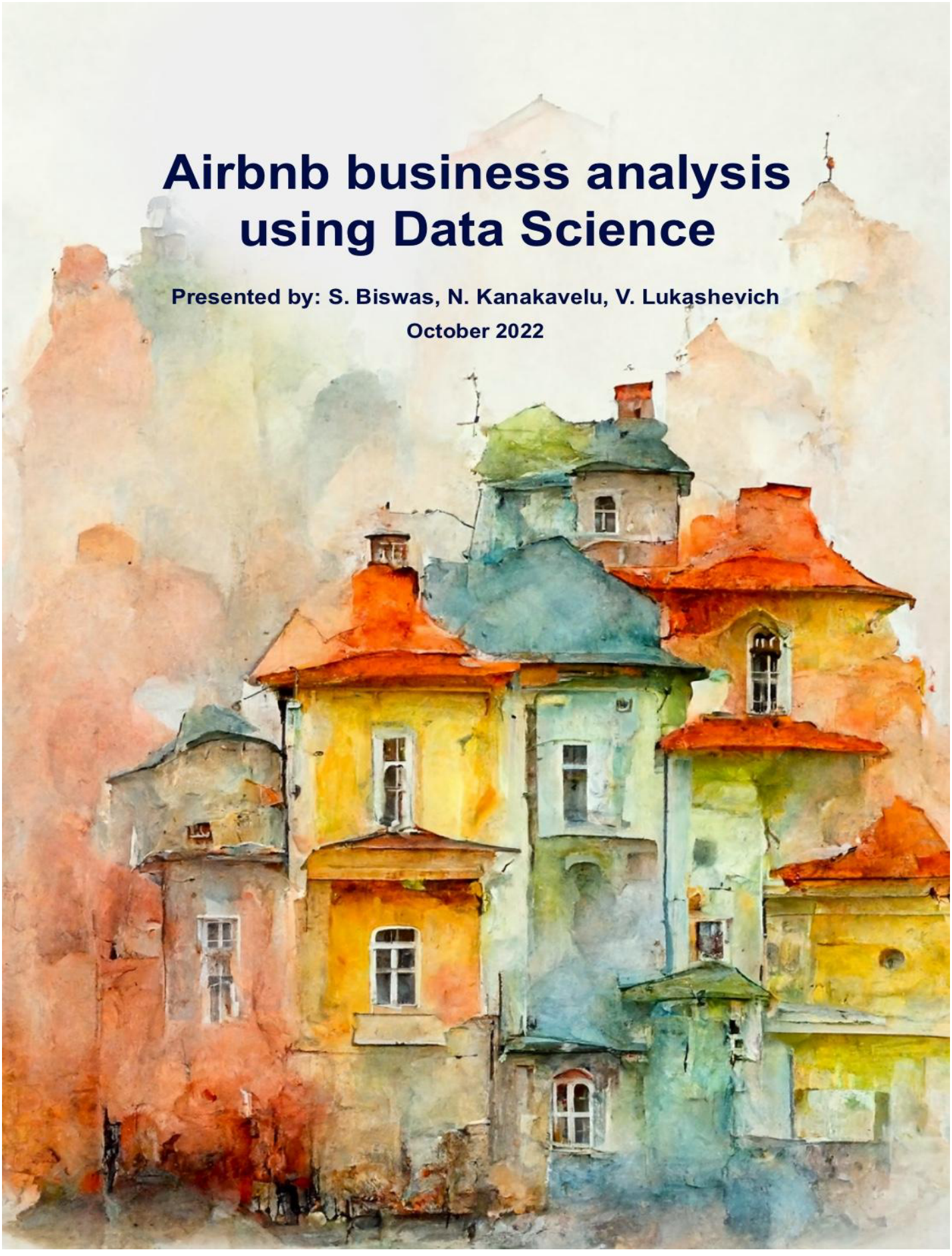
Airbnb business analysis using Data Science

Presented by: S. Biswas, N. Kanakavelu, V. Lukashevich

October 2022



Team Project
Airbnb business analysis using Data Science
October 2022
Version: 1.3

By

GROUP 1

Company: Airbnb
Industry Focus: Travel and hospitality experience for any trip
Founded: August 2008, San Francisco, California, United States

Version History:

No.	Date	Comments
1.0	09-Oct-2022	Initial team discussion about planning for the outline
1.1	20-Oct-2022	Understanding Airbnb data and business analytics requirements
1.2	27-Oct-2022	Discussing and updating notes from individual analysis to the document
1.3	29-Oct-2022	Final review and submission

Disclaimer: *The analysis is based on the dataset used from Kaggle.com for Airbnb and used as a study of Machine Learning as part of the degree course for M.Sc. in Artificial Technologies from the University of Essex.*

1 Table of Contents

2 Introduction

2019 was the last year before COVID-19 started. Tourism and the sharing economy appear to be the sectors most affected by the coronavirus. For example, in 2021, the number of listings has fallen in the U.S., however, people who rented preferred to stay longer: in New York, the average minimum days of stay in Airbnb service rose by 237% – from 6.1 to 20.6 (Kolomatsky, 2021). In 2022 the company returned to a profitable level. The present business analysis intends to remind us how the situation looked before the pandemic era for the smoothest way back to normal. Moreover, we assume that the list of the most attractive neighbourhoods to tourists in New York City could hardly have changed during the last few years.

3 Business Analytic Requirement

In this analysis we provide a system to predict competitive prices for Airbnb room listings based on locality in New York City. It could be useful not only for Airbnb executive board members, but also for developers, investors, real estate owners, rentiers, tourists, and businessmen.

4 Information (Data Source)

The analysis will be performed using the data from https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data

The data file is "AB_NYC_2019.csv" and load it into a data frame, ab_nyc_df to see how the dataset is formed.

```
import pandas as pd
ab_nyc_df = pd.read_csv("AB_NYC_2019.csv")
ab_nyc_df.info()
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 16 columns):
 #   Column                          Count Non-Null Dtype
---  ------                          -------- ----------   -----
 0   id                              48895 non-null  int64
 1   name                            48879 non-null  object
 2   host_id                         48895 non-null  int64
 3   host_name                       48874 non-null  object
 4   neighbourhood_group             48895 non-null  object
 5   neighbourhood                   48895 non-null  object
 6   latitude                        48895 non-null  float64
 7   longitude                       48895 non-null  float64
 8   room_type                       48895 non-null  object
 9   price                           48895 non-null  int64
 10  minimum_nights                  48895 non-null  int64
 11  number_of_reviews                       48895 non-null  int64
 12  last_review                     38843 non-null  object
 13  reviews_per_month               38843 non-null  float64
 14  calculated_host_listings_count  48895 non-null  int64
 15  availability_365                48895 non-null  int64
dtypes: float64(3), int64(7), object(6)
memory usage: 6.0+ MB
```

5 Exploratory Data Analysis (EDA)

From the data (above) we can see that there are 48,895 rows and 16 columns. More closely, we can see that there are columns that have missing data.

5.1 Understanding the data

5.1.1 List columns with NaN/Null data

To find out the columns and number of rows, we will use "isna" and "sum" to show the results (see below).

```
ab_nyc_df.isna().sum()
```

Output:

```
id                                0
name                             16
host_id                           0
host_name                        21
neighbourhood_group               0
neighbourhood                     0
latitude                          0
longitude                         0
room_type                         0
price                             0
minimum_nights                    0
number_of_reviews                 0
last_review                   10052
reviews_per_month             10052
calculated_host_listings_count    0
availability_365                  0
dtype: int64
```

The columns with missing data are: name, host_name, last_review and reviews_per_month.

5.1.2 List columns regions (Neighbourhood Group)

Now let's try to understand the data held within "neighbourhood_group" and "neighbourhood".

```
ab_nyc_df['neighbourhood_group'].unique()
```

Output:

```
array(['Brooklyn', 'Manhattan', 'Queens', 'Staten Island', 'Bronx'],
      dtype=object)
```

This gives us 5 regions within New York listings.

5.1.3 List columns planning areas (Neighbourhood)

Let's see how these regions are further distributed within planning areas.

```
ab_nyc_df['neighbourhood'].unique()
```

Output:

```
array(['Kensington', 'Midtown', 'Harlem', 'Clinton Hill', 'East Harlem',
       'Murray Hill', 'Bedford-Stuyvesant', "Hell's Kitchen",
       'Upper West Side', 'Chinatown', 'South Slope', 'West Village',
       'Williamsburg', 'Fort Greene', 'Chelsea', 'Crown Heights',
       'Park Slope', 'Windsor Terrace', 'Inwood', 'East Village',
       'Greenpoint', 'Bushwick', 'Flatbush', 'Lower East Side',
       'Prospect-Lefferts Gardens', 'Long Island City', 'Kips Bay',
       'SoHo', 'Upper East Side', 'Prospect Heights',
       'Washington Heights', 'Woodside', 'Brooklyn Heights',
       'Carroll Gardens', 'Gowanus', 'Flatlands', 'Cobble Hill',
       'Flushing', 'Boerum Hill', 'Sunnyside', 'DUMBO', 'St. George',
       'Highbridge', 'Financial District', 'Ridgewood',
       'Morningside Heights', 'Jamaica', 'Middle Village', 'NoHo',
       'Ditmars Steinway', 'Flatiron District', 'Roosevelt Island',
       'Greenwich Village', 'Little Italy', 'East Flatbush',
       'Tompkinsville', 'Astoria', 'Clason Point', 'Eastchester',
       'Kingsbridge', 'Two Bridges', 'Queens Village', 'Rockaway Beach',
       'Forest Hills', 'Nolita', 'Woodlawn', 'University Heights',
       'Gravesend', 'Gramercy', 'Allerton', 'East New York',
       'Theater District', 'Concourse Village', 'Sheepshead Bay',
       'Emerson Hill', 'Fort Hamilton', 'Bensonhurst', 'Tribeca',
       'Shore Acres', 'Sunset Park', 'Concourse', 'Elmhurst',
       'Brighton Beach', 'Jackson Heights', 'Cypress Hills', 'St. Albans',
       'Arrochar', 'Rego Park', 'Wakefield', 'Clifton', 'Bay Ridge',
       'Graniteville', 'Spuyten Duyvil', 'Stapleton', 'Briarwood',
       'Ozone Park', 'Columbia St', 'Vinegar Hill', 'Mott Haven',
       'Longwood', 'Canarsie', 'Battery Park City', 'Civic Center',
       'East Elmhurst', 'New Springville', 'Morris Heights', 'Arverne',
       'Cambria Heights', 'Tottenville', 'Mariners Harbor', 'Concord',
       'Borough Park', 'Bayside', 'Downtown Brooklyn', 'Port Morris',
       'Fieldston', 'Kew Gardens', 'Midwood', 'College Point',
       'Mount Eden', 'City Island', 'Glendale', 'Port Richmond',
       'Red Hook', 'Richmond Hill', 'Bellerose', 'Maspeth',
       'Williamsbridge', 'Soundview', 'Woodhaven', 'Woodrow',
       'Co-op City', 'Stuyvesant Town', 'Parkchester', 'North Riverdale',
       'Dyker Heights', 'Bronxdale', 'Sea Gate', 'Riverdale',
       'Kew Gardens Hills', 'Bay Terrace', 'Norwood', 'Claremont Village',
       'Whitestone', 'Fordham', 'Bayswater', 'Navy Yard', 'Brownsville',
       'Eltingville', 'Fresh Meadows', 'Mount Hope', 'Lighthouse Hill',
       'Springfield Gardens', 'Howard Beach', 'Belle Harbor',
       'Jamaica Estates', 'Van Nest', 'Morris Park', 'West Brighton',
       'Far Rockaway', 'South Ozone Park', 'Tremont', 'Corona',
       'Great Kills', 'Manhattan Beach', 'Marble Hill', 'Dongan Hills',
       'Castleton Corners', 'East Morrisania', 'Hunts Point', 'Neponsit',
       'Pelham Bay', 'Randall Manor', 'Throgs Neck', 'Todt Hill',
       'West Farms', 'Silver Lake', 'Morrisania', 'Laurelton',
       'Grymes Hill', 'Holliswood', 'Pelham Gardens', 'Belmont',
       'Rosedale', 'Edgemere', 'New Brighton', 'Midland Beach',
```

```
'Baychester', 'Melrose', 'Bergen Beach', 'Richmondtown',
'Howland Hook', 'Schuylerville', 'Coney Island', 'New Dorp Beach',
"Prince's Bay", 'South Beach', 'Bath Beach', 'Jamaica Hills',
'Oakwood', 'Castle Hill', 'Hollis', 'Douglaston', 'Huguenot',
'Olinville', 'Edenwald', 'Grant City', 'Westerleigh',
'Bay Terrace, Staten Island', 'Westchester Square', 'Little Neck',
'Fort Wadsworth', 'Rosebank', 'Unionport', 'Mill Basin',
'Arden Heights', "Bull's Head", 'New Dorp', 'Rossville',
'Breezy Point', 'Willowbrook'], dtype=object)
```

```
ab_nyc_df['neighbourhood'].nunique()
```

Output:

221

So, there are 221 further areas distributed within 5 regions.

5.1.4 List columns room types

Now, let's understand the types of rooms provided by the listings.

```
ab_nyc_df['room_type'].unique()
```

Output:

```
array(['Private room', 'Entire home/apt', 'Shared room'], dtype=object)
```

5.2 Cleansing

Now, we will clean the data as required by us for the analysis i.e. removing columns that are not required and then replacing any NaN (null) values.

Based on the analysis required, let's drop columns that are not required: id, host_name, and last_review. We have dropped 'host_name' not only because it is insignificant but also for ethical reasons.

5.2.1 Drop columns

```
ab_nyc_df.drop(['id','host_name','last_review'], axis=1, inplace=True)

ab_nyc_df.isna().sum()
```

Output:

```
host_id                         0
neighbourhood_group             0
neighbourhood                   0
latitude                        0
longitude                       0
room_type                       0
price                           0
```

```
minimum_nights                  0
number_of_reviews               0
reviews_per_month               10052
calculated_host_listings_count  0
availability_365                0
dtype: int64
```

5.2.2 Update NaN values to 0

Seeing the final table (above), now we are left with "reviews_per_month" having NaN values. Let's replace it with 0.

```
ab_nyc_df.reviews_per_month.fillna(0, inplace=True)

ab_nyc_df.isna().sum()
```

Output:

```
host_id                         0
neighbourhood_group             0
neighbourhood                   0
latitude                        0
longitude                       0
room_type                       0
price                           0
minimum_nights                  0
number_of_reviews               0
reviews_per_month               0
calculated_host_listings_count  0
availability_365                0
dtype: int64
```

So, no missing data and we should be good to go-ahead.

5.3 Statistical Analysis

Now that we have the data frame required for further exploration, we will start working on various relations within the data.

We will try to visualize and understand the listing distributions in relation to room types and average pricing.

5.3.1 Libraries used for exploration and visualisation

matplotlib: from https://matplotlib.org/, for visualisation

matplotlib.pyplot: from https://matplotlib.org/stable/tutorials/introductory/pyplot.html, for plotting

numpy: from https://numpy.org/, for scientific computing

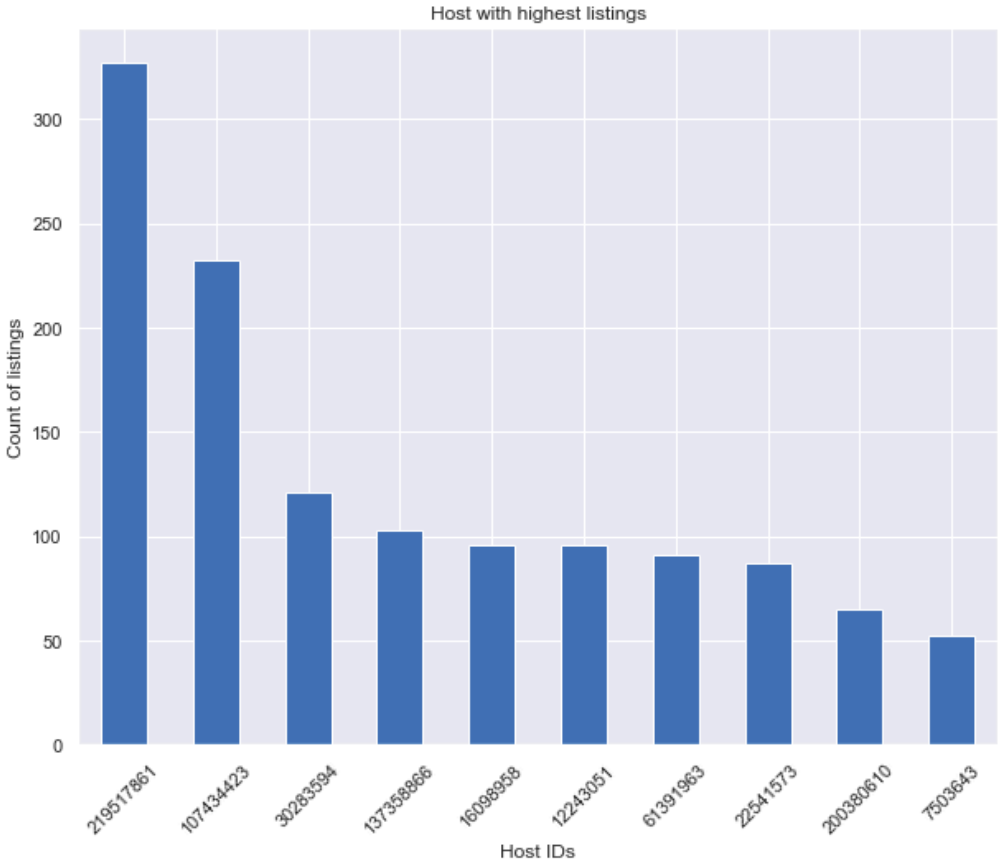
seaborn: from https://seaborn.pydata.org/, for statistical data visualisations

5.3.2 Host with highest listings (Top 10)

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import seaborn as sns
import matplotlib.cbook as cbook

top_host_id = ab_nyc_df['host_id'].value_counts().head(10)
sns.set(rc={'figure.figsize':(10,8)})
viz_bar = top_host_id.plot(kind='bar')
viz_bar.set_title('Host with highest listings')
viz_bar.set_xlabel('Host IDs')
viz_bar.set_ylabel('Count of listings')
viz_bar.set_xticklabels(viz_bar.get_xticklabels(), rotation=45)
```

Output:

```
[Text(0, 0, '219517861'),
 Text(1, 0, '107434423'),
 Text(2, 0, '30283594'),
 Text(3, 0, '137358866'),
 Text(4, 0, '16098958'),
 Text(5, 0, '12243051'),
 Text(6, 0, '61391963'),
 Text(7, 0, '22541573'),
 Text(8, 0, '200380610'),
 Text(9, 0, '7503643')]
```



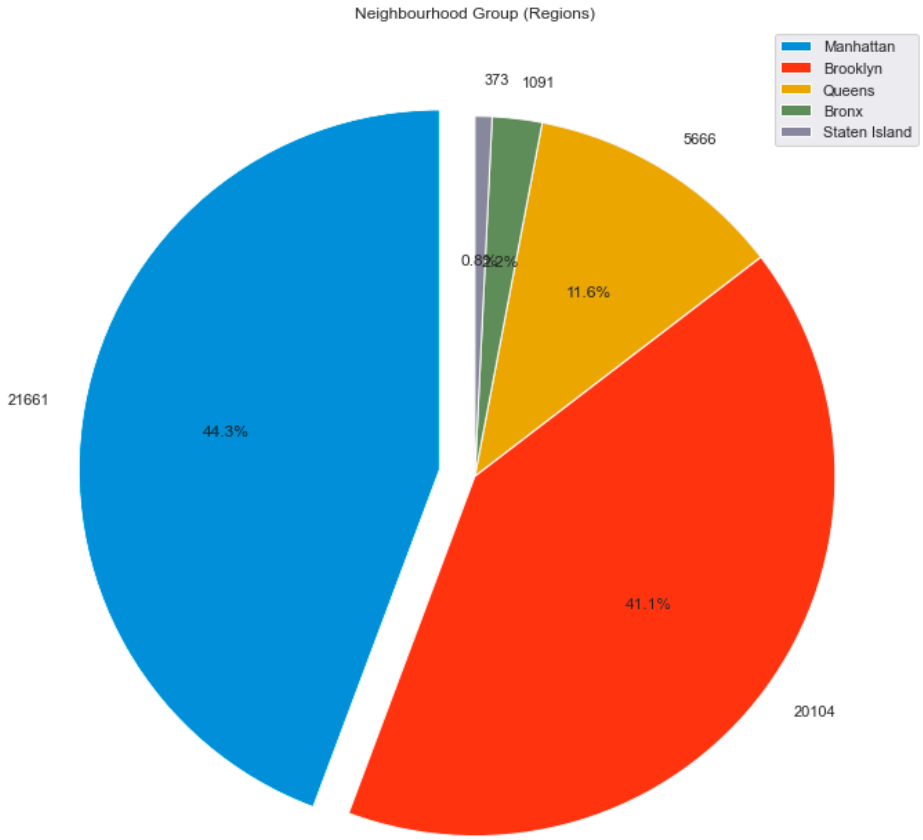
So, out of 37457 listings we can see that 1270 constitutes the top 10 of which there is a list of more than 350.

5.3.3 Top listings in the regions

Here, let's find how each region ("neighbourhood_group') is held by the listings.

```
labels = ab_nyc_df.neighbourhood_group.value_counts().index
colors = ['#008fd6','#fc4f31','#e5ae39','#6d905f','#8b8b9b']
explode = (0.1,0,0,0,0)

shape = ab_nyc_df.neighbourhood_group.value_counts().values

plt.figure(figsize=(12,12))
plt.pie(shape, explode = explode, labels=shape, colors= colors, autopct = '%1.1f%%',
startangle=90)

plt.legend(labels)
plt.title('Neighbourhood Group (Regions)')

plt.show()
```

Output:



The chart shows that Manhattan is the highest but Brooklyn being the next together sums for about 85% of the total listings.

5.3.4 Top 10 planning areas

```
ab_nyc_df.neighbourhood.value_counts().head(10)
```

Output:

```
Williamsburg            3920
Bedford-Stuyvesant      3714
Harlem                  2658
Bushwick                2465
Upper West Side         1971
Hell's Kitchen          1958
East Village            1853
Upper East Side         1798
Crown Heights           1564
Midtown                 1545
Name: neighbourhood, dtype: int64
```

5.3.5 Coordinates / Locations

We will be required to draw out the coordinates so that the regions can be plotted into the map.

```
coord = ab_nyc_df.loc[:,['longitude','latitude']]
coord.describe()
```

Output:

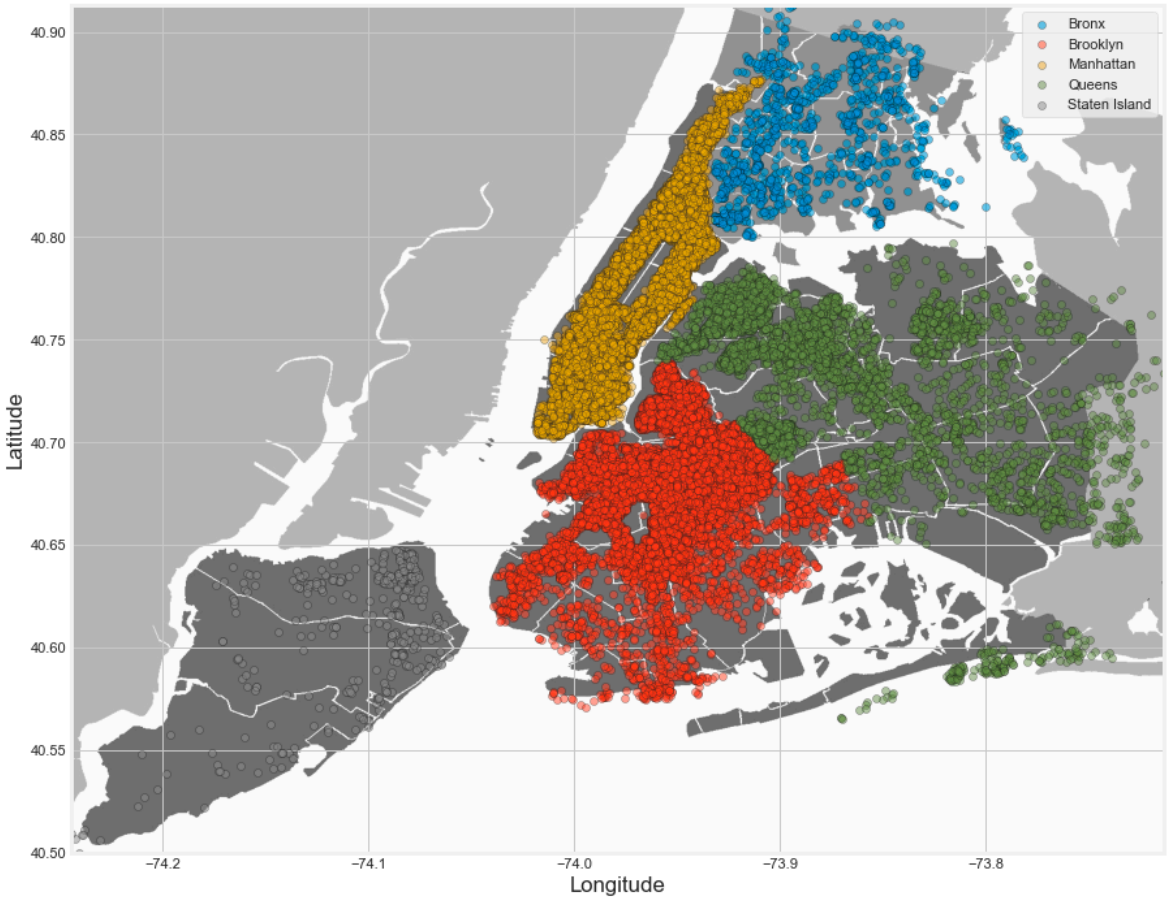
	longitude	latitude
count	48895.000000	48895.000000
mean	-73.952170	40.728949
std	0.046157	0.054530
min	-74.244420	40.499790
25%	-73.983070	40.690100
50%	-73.955680	40.723070
75%	-73.936275	40.763115
max	-73.712990	40.913060

5.3.6 Regions on New York Map

So, from the New York map we can see that the denser listings are shown in red (Brooklyn) followed by yellow (Manhattan). The listings decrease more towards the east and northern regions but more towards the western regions.

```
plt.figure(figsize=(18,12))
plt.style.use('fivethirtyeight')

# using the min and max of longitude and latitude
BBox = (-74.244420, -73.712990, 40.499790, 40.913060)

sg_map = mpimg.imread('New_York_City1.png')

plt.imshow(sg_map,zorder=0,extent=BBox)
ax = plt.gca()
groups = ab_nyc_df.groupby('neighbourhood_group')

for name,group in groups :
    plt.scatter(group['longitude'],group['latitude'],label=name,alpha=0.5, edgecolors='k')

    plt.xlabel('Longitude')
    plt.ylabel('Latitude')
    plt.legend()
```

Output:

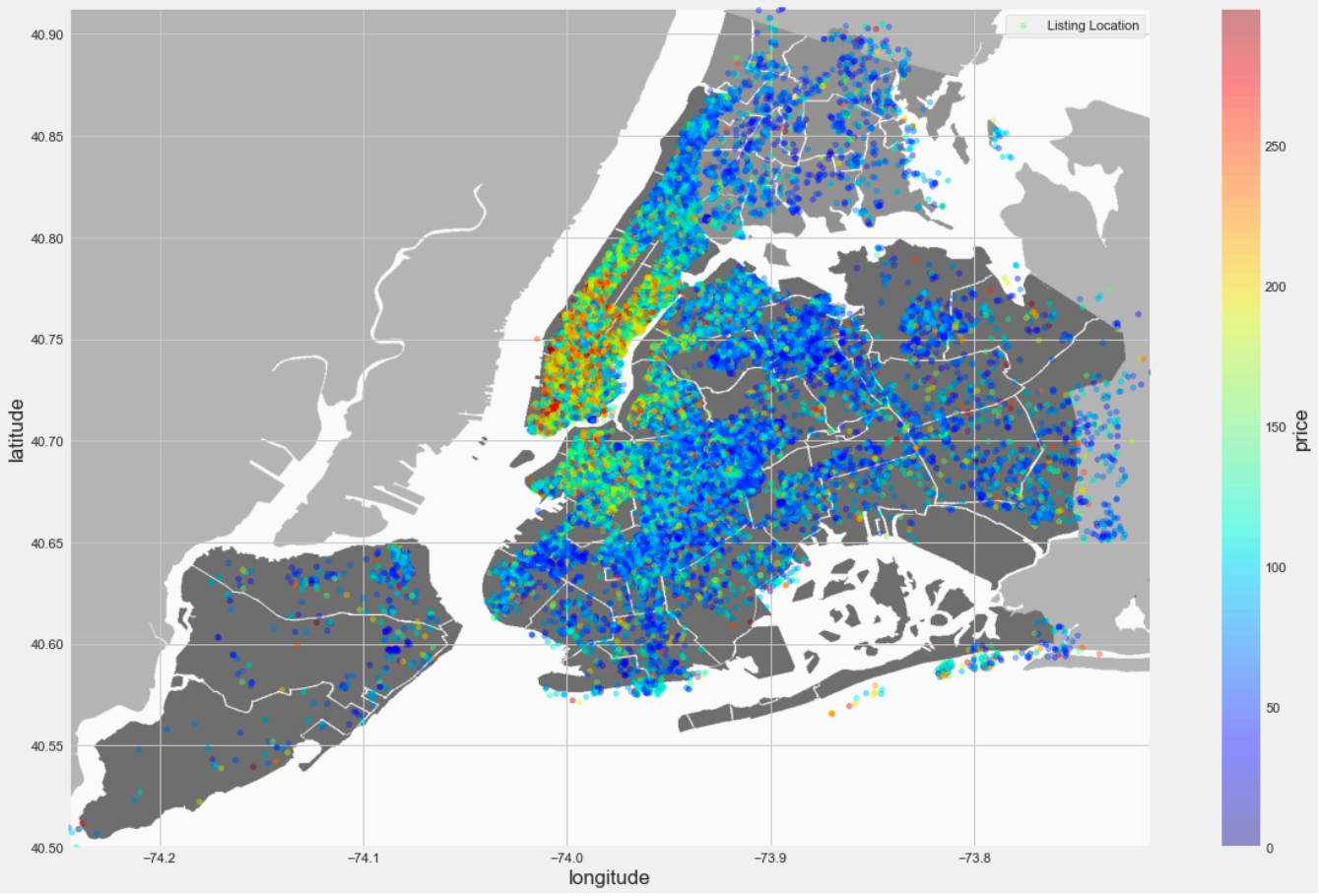


5.3.7 Add price to the region map

The price heatmap shows that the price goes up towards the Manhattan and Brooklyn regions of New York.

```
ab_nyc_df_1 = ab_nyc_df[ab_nyc_df.price < 300]

plt.figure(figsize=(18,12))

sg_map = mpimg.imread('New_York_City1.png')

plt.imshow(sg_map,zorder=0,extent=BBox)

ax = plt.gca()

ab_nyc_df_1.plot(kind='scatter',x='longitude',y='latitude',label='Listing Location',
c='price', ax=ax, cmap=plt.get_cmap('jet'), colorbar=True, alpha=0.4, zorder=5)

plt.legend()
plt.show()
```

Output:

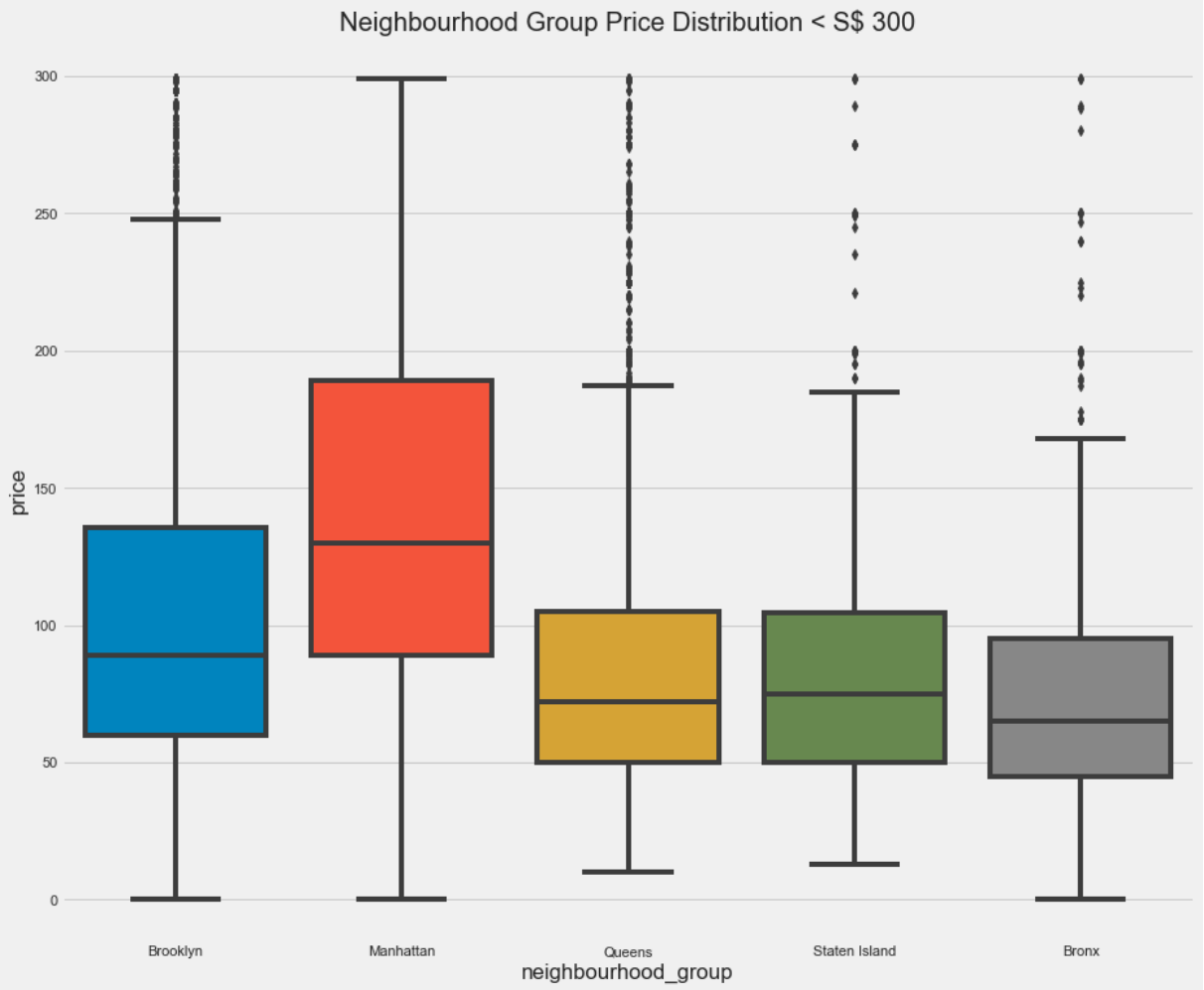


5.3.8 Price Distribution

After reviewing the price heat map, let's try to understand how the price is distributed using a box-plot. This will give us information about the price range grouped by "neighbourhood_group"/region area.

```
plt.style.use('fivethirtyeight')
plt.figure(figsize=(14,12))
sns.boxplot(y='price',x='neighbourhood_group',data = ab_nyc_df_1)
plt.title('Neighbourhood Group Price Distribution < S$ 300')
plt.show()
```

Output:



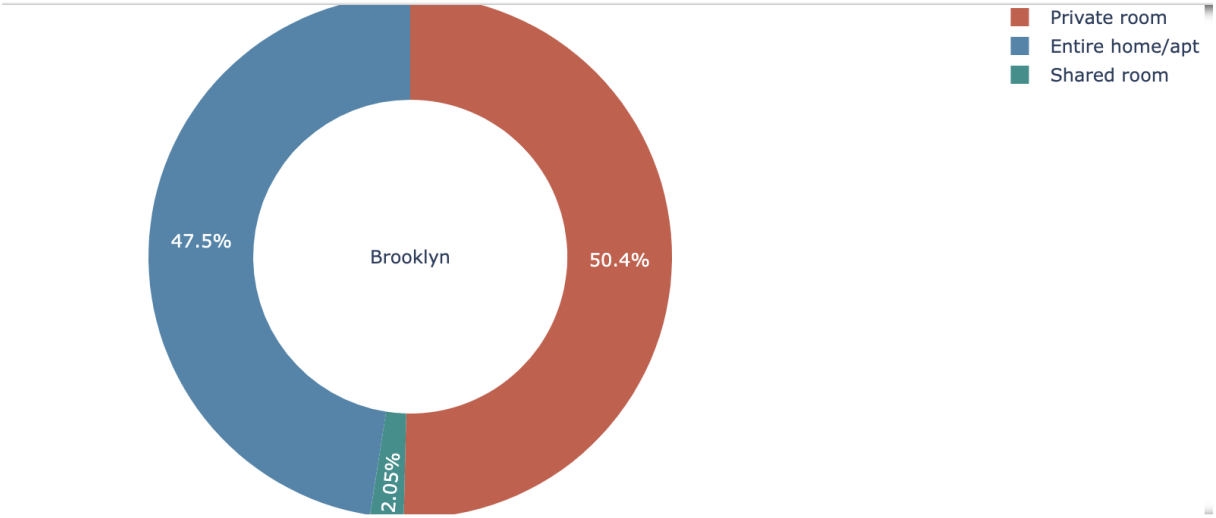
This also confirms that the Manhattan region has the most expensive price per night with a median of approximately $135.

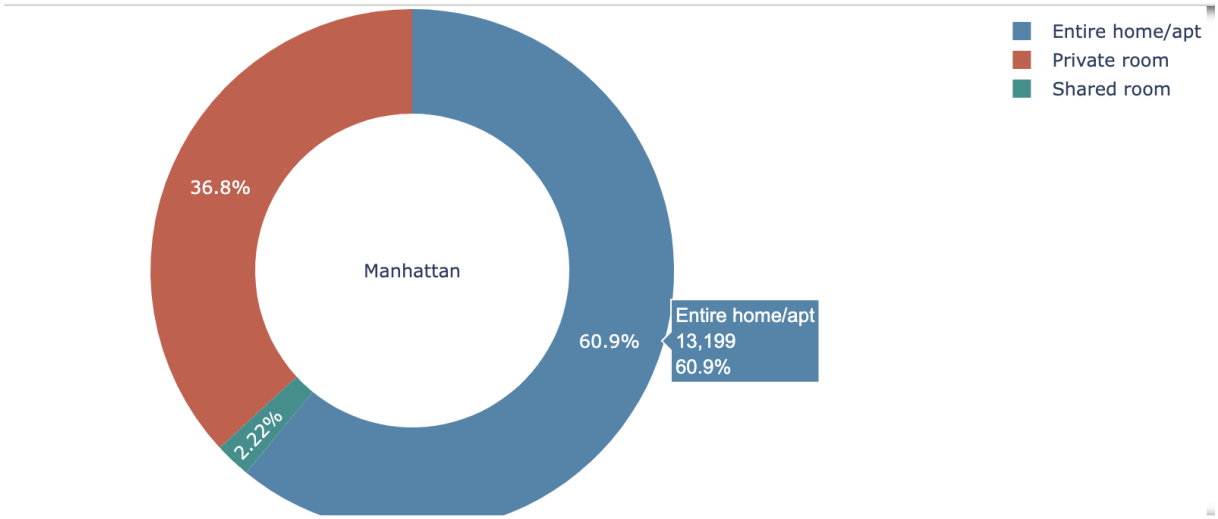
5.3.9 Room Types for Listings

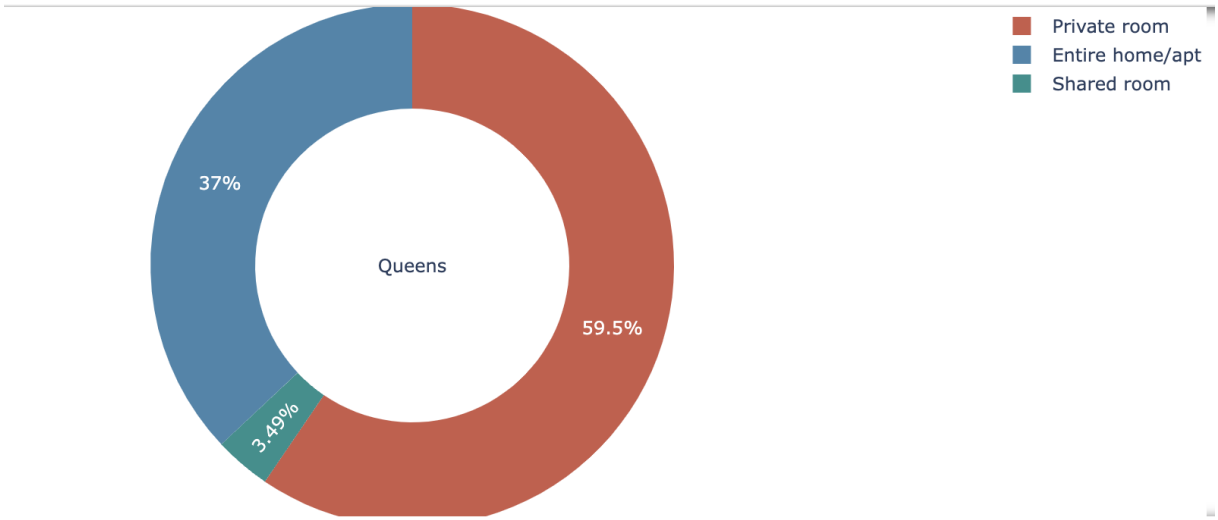
From this we see that Brooklyn and Manhattan are the only regions that show a majority of as "Entire home/apt" with others dominated by "Private room types".

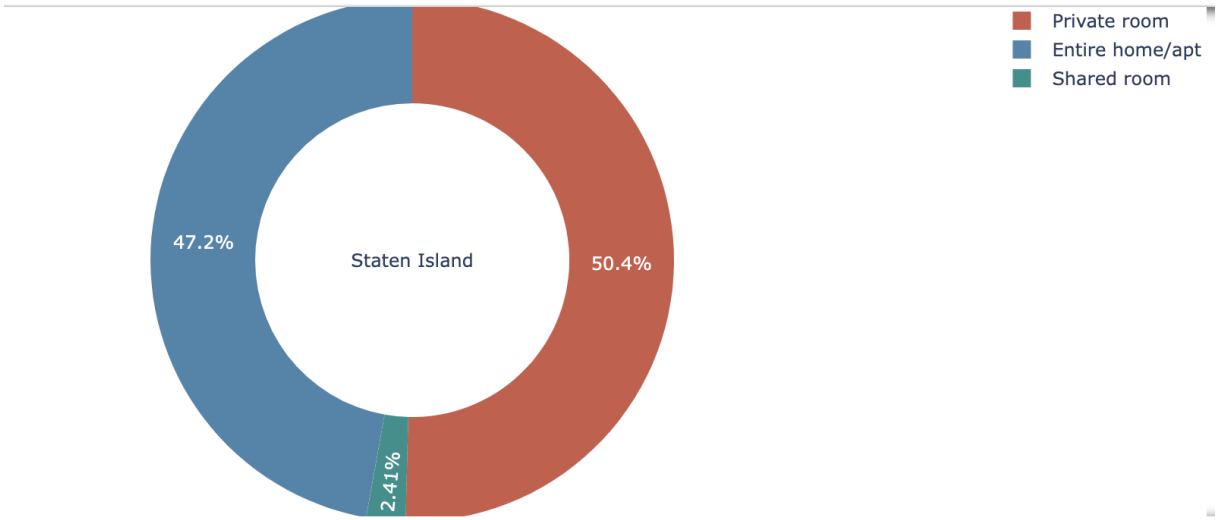
```
import plotly.offline as pyo
import plotly.graph_objs as go

# set colors for 'Private room', 'Entire home/apt', 'Shared room'
color_dict = {'Private room': '#cc5a49', 'Entire home/apt' : '#4586ac', 'Shared room' :
'#21908d'}

#Group the room type using 'neighbourhood_group' as an index
airbnb_types=ab_nyc_df.groupby(['neighbourhood_group', 'room_type']).size()

#Create function to plot room type proportion on all region area
for region in ab_nyc_df.neighbourhood_group.unique():
    plt.figure(figsize=(24,12))

    airbnb_reg=airbnb_types[region]
    labels = airbnb_reg.index
    sizes = airbnb_reg.values

    colors = [color_dict[x] for x in labels]

    plot_num = 321
    plt.subplot(plot_num)
    reg_ch = go.Figure(data = [go.Pie(labels = labels, values = sizes, hole = 0.6)])
    reg_ch.update_traces(title = region, marker=dict(colors=colors))
    reg_ch.show()

    plot_num += 1
```

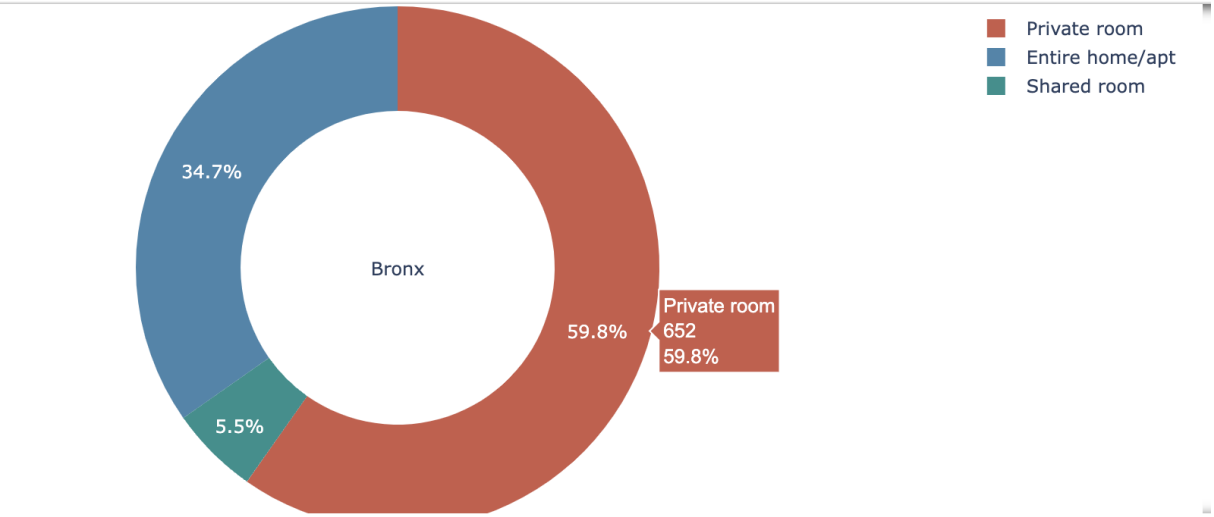
Output:




Entire home/apt
Private room
Shared room
36.8%
Manhattan
60.9%
Entire home/apt
13,199
60.9%
2.22%


Private room
Entire home/apt
Shared room
37%
Queens
59.5%
3.49%


Private room
Entire home/apt
Shared room
47.2%
Staten Island
50.4%
2.41%



5.3.10 Most Reviewed Listings (Top 10)

Within the top 10 popular listings we see that along with Manhattan and Brooklyn, Queens also has positions. But supporting our finding we still see that the majority is still covered by Manhattan.

```
ab_nyc_df.nlargest(10, 'number_of_reviews')
```

Output:

Out[40]:

name	host_id	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	reviews_per_mor
oom near JFK Queen Bed	47621202	Queens	Jamaica	40.66730	-73.76831	Private room	47	1	629	14.
Great Bedroom in Manhattan	4734398	Manhattan	Harlem	40.82085	-73.94025	Private room	49	1	607	7.
Beautiful Bedroom in Manhattan	4734398	Manhattan	Harlem	40.82124	-73.93838	Private room	49	1	597	7.
Private Bedroom in Manhattan	4734398	Manhattan	Harlem	40.82264	-73.94041	Private room	49	1	594	7.
Room Near JFK Twin Beds	47621202	Queens	Jamaica	40.66939	-73.76975	Private room	47	1	576	13.
Steps away from Laguardia airport	37312959	Queens	East Elmhurst	40.77006	-73.87683	Private room	46	1	543	11.
Manhattan Lux t.Like.Love.Lots.Look !	2369681	Manhattan	Lower East Side	40.71921	-73.99116	Private room	99	2	540	6.
Cozy Room Family lome LGA Airport NO CLEANING FEE	26432133	Queens	East Elmhurst	40.76335	-73.87007	Private room	48	1	510	16.
Private brownstone studio Brooklyn	12949460	Brooklyn	Park Slope	40.67926	-73.97711	Entire home/apt	160	1	488	8.
Private Room/Family Friendly	792159	Brooklyn	Bushwick	40.70283	-73.92131	Private room	60	3	480	6.

5.3.11 Average Price Per Night

Let's find the average price for the listings based on top 10 reviews.

```
top_review =ab_nyc_df.nlargest(10, 'number_of_reviews')
price_avg = top_review.price.mean()
print('Average price per night: ${}'.format(price_avg))
```

Output:

Average price per night: $65.4

This shows that there is an average price of $65 with 6/10 listings are under $60 and they are of "Private Room" type.

1.

5.4 EDA - Outcome

By doing a gradual EDA on the dataset, it reveals how the various region listings are distributed in New York along with how their spread is seen within different locations. It shows how Manhattan followed by Brooklyn dominates the listings and its relationship with the price in nearby regions and how this form a selling value.

3.

4. Correlation Heat Map

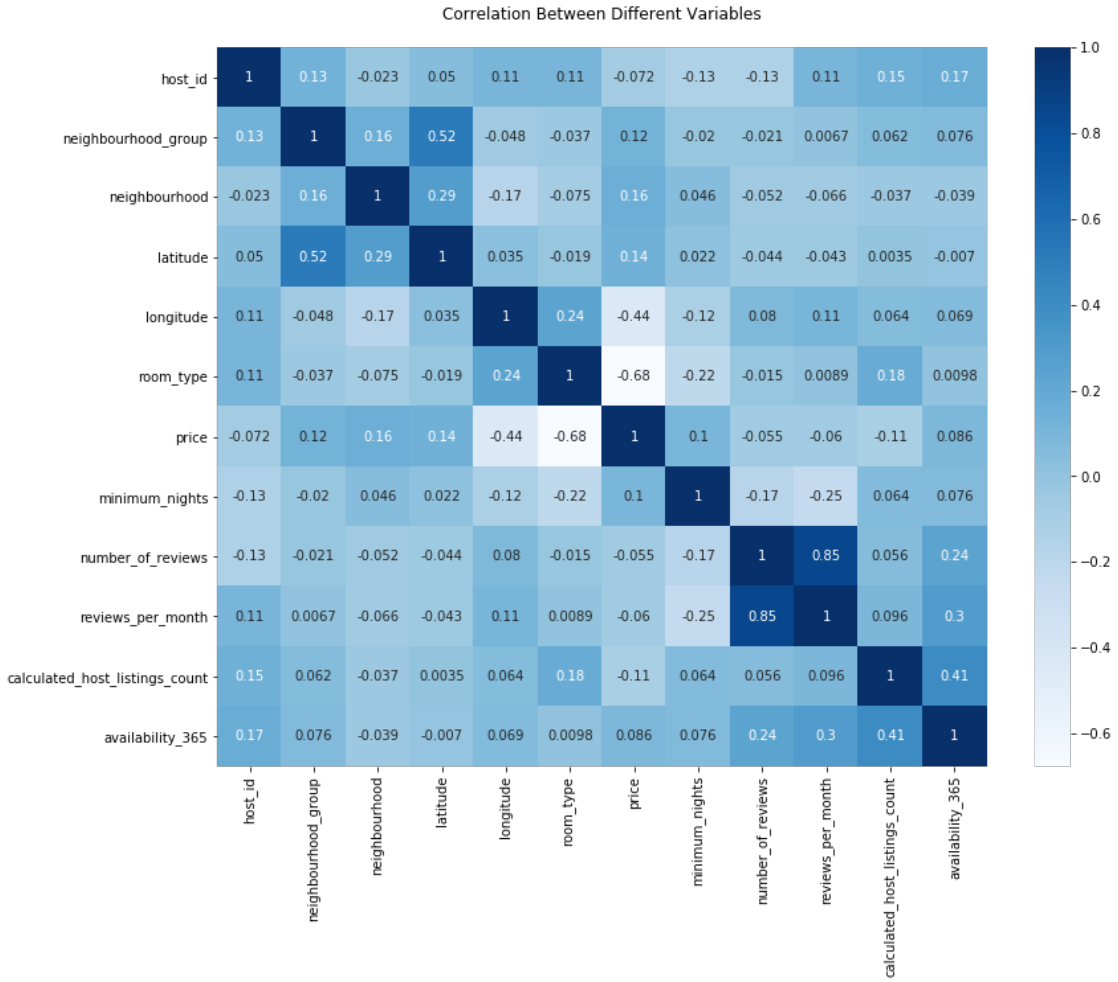
Preprocessing to encode labels for categorical type variables

```
from sklearn import preprocessing

#encode label
encode = preprocessing.LabelEncoder()
encode.fit(ab_nyc_df.neighbourhood_group)
ab_nyc_df.neighbourhood_group=encode.transform(ab_nyc_df.neighbourhood_group)

encode = preprocessing.LabelEncoder()
encode.fit(ab_nyc_df.neighbourhood)
ab_nyc_df.neighbourhood=encode.transform(ab_nyc_df.neighbourhood)

encode = preprocessing.LabelEncoder()
encode.fit(ab_nyc_df.room_type)
ab_nyc_df.room_type=encode.transform(ab_nyc_df.room_type)

ab_nyc_df.sort_values(by='price',ascending=True,inplace=True)
```

Spearman correlation coefficient was calculated and presented as heatmap to visualise the correlation between different variables.

```
corr = ab_nyc_df.corr(method='spearman')
plt.figure(figsize=(13,10))
plt.title("Correlation Between Different Variables\n")
sns.heatmap(corr, annot=True, cmap='Blues')
plt.show()
```

Output:



From the above correlation map, it is clear that the room type and location of the room (longitude) have a better correlation with the price compared to other variables.

6 Linear Relationship

Multivariate linear regression model was built to predict the dependent variable ‘Price’ from the independent variables 'neighbourhood_group', 'neighbourhood', 'latitude', longitude' and 'room_type’

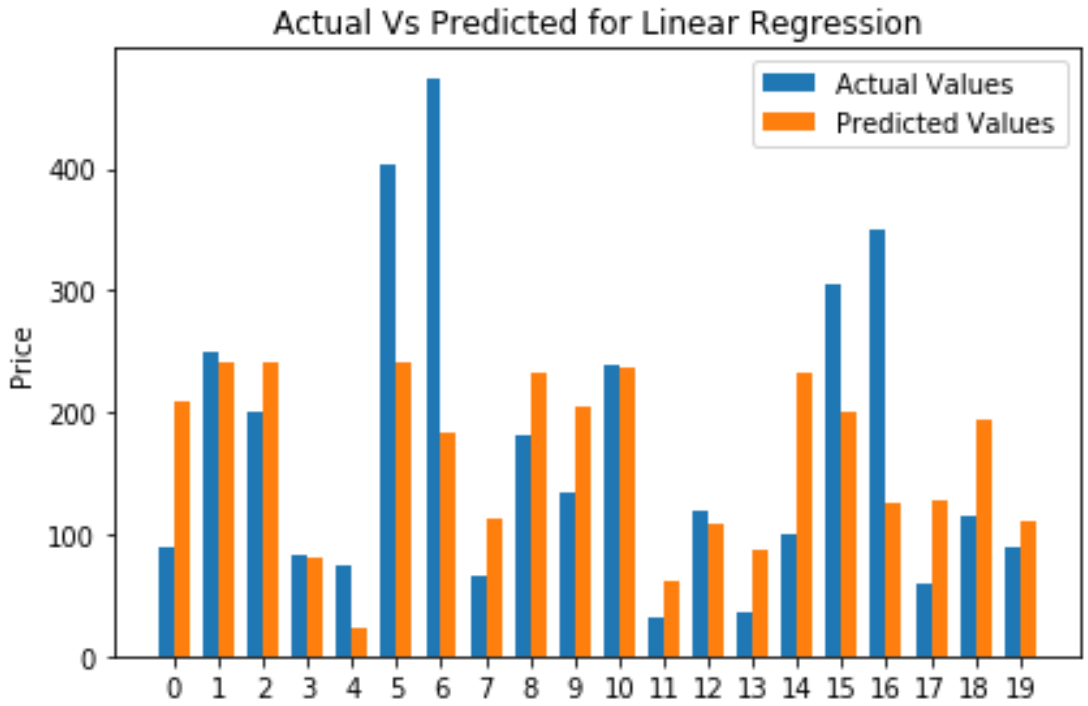
```
import pandas as pd
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
from sklearn.metrics import mean_squared_error,r2_score, mean_absolute_error

X =
ab_nyc_df[['neighbourhood_group','neighbourhood','latitude','longitude','room_type']]
y = ab_nyc_df['price']

lin_reg = LinearRegression()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
lin_reg.fit(X_train,y_train)

predict_LR = lin_reg.predict(X_test)

print("Mean Squared Error: ", np.sqrt(metrics.mean_squared_error(y_test, predict_LR)))
print("R2 Score: ", r2_score(y_test,predict_LR) * 100)
print("Mean Absolute Error: ", mean_absolute_error(y_test,predict_LR))
print("Mean Squareroot Error: ", mean_squared_error(y_test,predict_LR))

#Actual Vs Predicted for Linear Regression
LR_pred_df = pd.DataFrame({
        'actual_values': np.array(y_test).flatten(),
        'predicted_values': predict_LR.flatten()}).head(20)

print(LR_pred_df.head(5))

x = LR_pred_df.index
width = 0.35
fig, ax = plt.subplots()
rects1 = ax.bar(x - width/2, LR_pred_df.actual_values, width, label='Actual Values')
rects2 = ax.bar(x + width/2, LR_pred_df.predicted_values, width, label='Predicted
Values')
ax.set_ylabel('Price')
ax.set_title('Actual Vs Predicted for Linear Regression')
ax.set_xticks(x)
ax.legend()
fig.tight_layout()
plt.show()
```

Output:

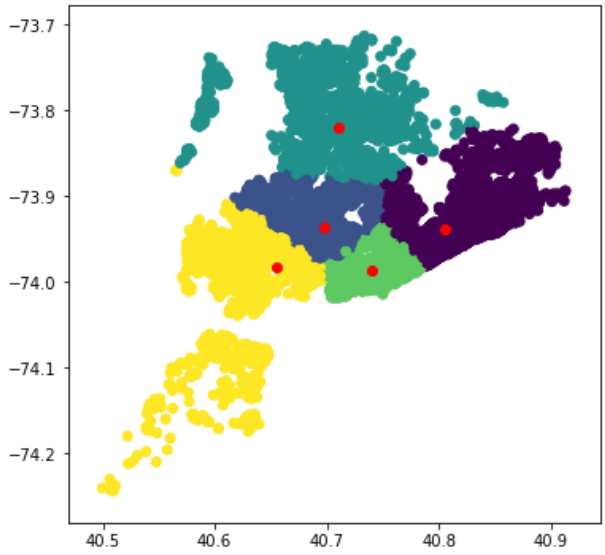
Model assessment results

Mean Squared Error: 200.51002576590074
R2 Score: 10.144070321165511
Mean Absolute Error: 74.97074301064022
Mean Squareroot Error: 40204.27043264217

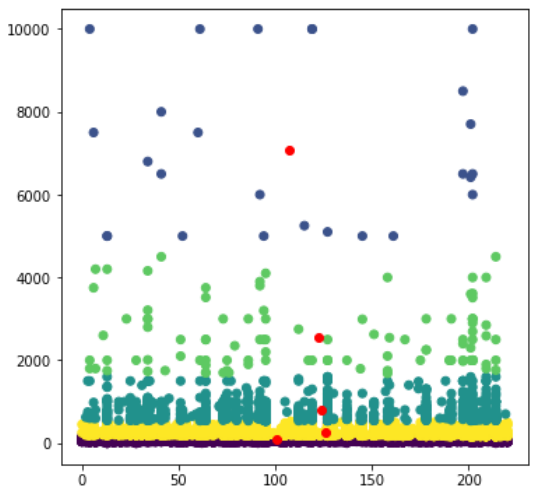


7 Clustering

```
data = np.array(list(ab_nyc_df[['latitude','longitude']].apply(tuple, axis=1)))
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=5, random_state=0).fit(data)
kmeans.labels_
kmeans.cluster_centers_
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
fig = plt.figure(figsize=(6,6))
ax = fig.add_subplot(1, 1, 1)
ax.scatter(data[:,0],data[:,1], c = kmeans.labels_)
fig.canvas.draw()

ax.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],c='red')

fig.canvas.draw()
```




```
data = np.array(list(ab_nyc_df[['neighbourhood','price']].apply(tuple, axis=1)))
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=5, random_state=0).fit(data)
kmeans.labels_
kmeans.cluster_centers_
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
fig = plt.figure(figsize=(6,6))
ax = fig.add_subplot(1, 1, 1)
ax.scatter(data[:,0],data[:,1], c = kmeans.labels_)
fig.canvas.draw()

ax.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],c='red')

fig.canvas.draw()
```



8 Analytical Report

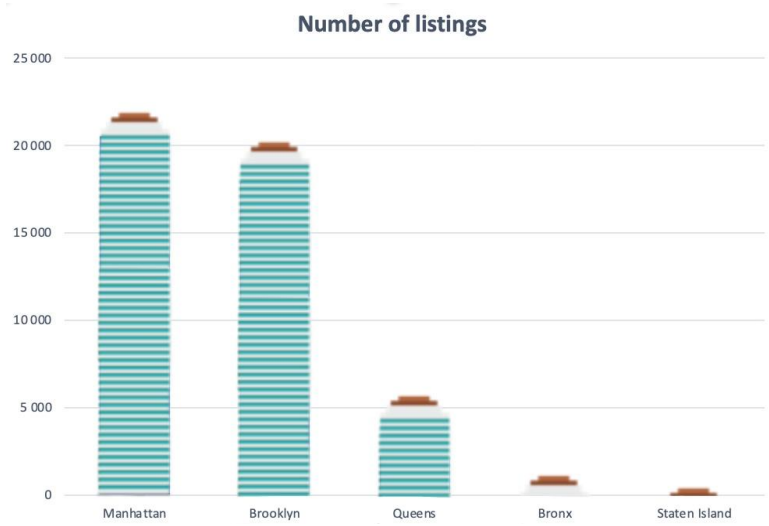
In this part we prepared the summary of our findings in an easy-to-use presentation manner.

According to the dataset, there are 5 regions in New York City with 221 neighbourhoods, however we found a large gap between them in terms of popularity for renting on Airbnb.

Manhattan and Brooklyn are the most travelled destinations.

Interpreting the number of listings, we can see that Manhattan is in the first position, depicted with the highest skyscraper. Then Brooklyn goes with the almost the same tall indicator level.

Together they sum for about 85% of the total listings: 44.3% of them are located in Manhattan, 41.1% ---– in Brooklyn.Queens is in the third position but it has a much lower number of listings – 11.6%, Bronx and Staten Island look very low.

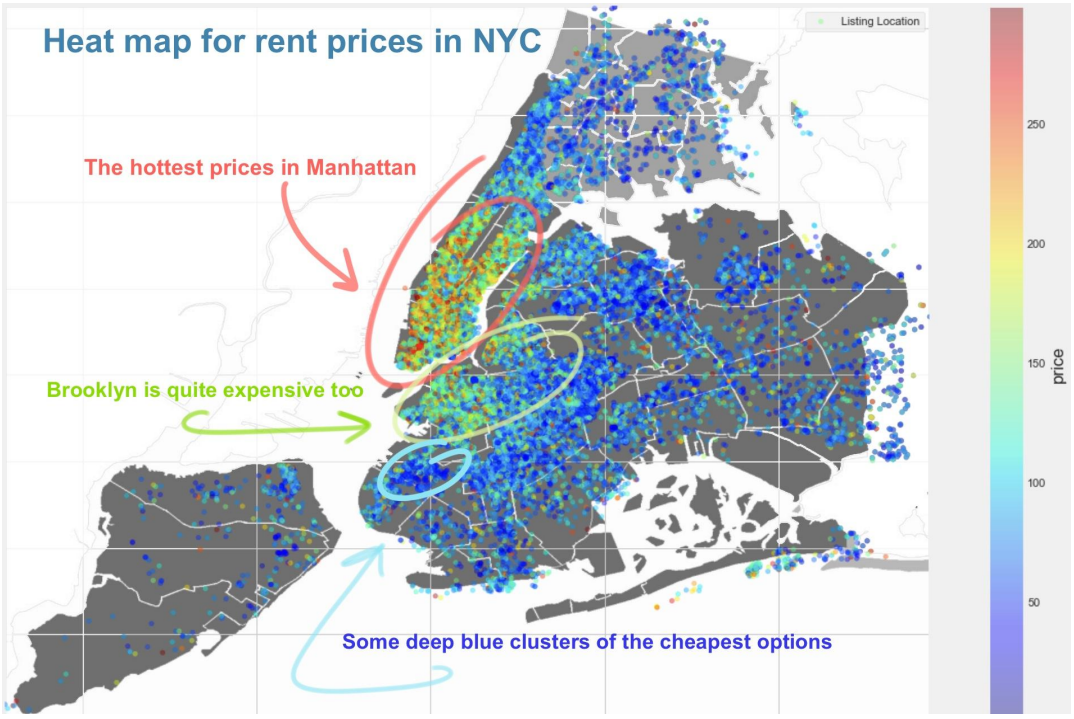


New York is the most important city of all in the U.S. considering the revenue-generating and income for Airbnb (Dudás et al., 2017).

Using the NYC map and the heatmap of prices we can see that the Manhattan region has the most expensive price per night with a median of approximately $135.

The average price in the whole dataset is $65 with 6/10 listings are under $60 and they are private rooms.

The highest density of renting is in Brooklyn, followed by Manhattan.



According to (Lehr, 2015), the average cost for a hotel room in NYC is almost 2,5 times more expensive than Airbnb shared room and about 30% higher than the Airbnb apartment cost.

We visualised the listing distributions in relation to room types.

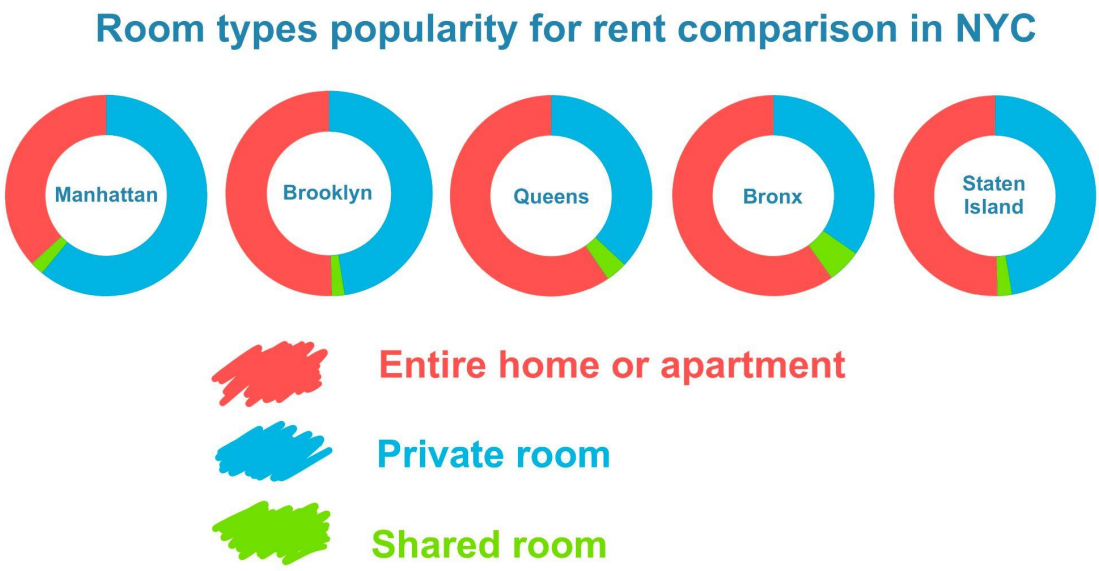
From the next illustration we see that Brooklyn and Manhattan are the only regions that show a majority of as "Entire home/apt" with others dominated by "Private room types".

The most popular rent type in Manhattan is renting a whole house or apartment – almost 61%, about 37% covered by private rooms.

The situation in Queens is mirrored: 37% for the entire apartment and 59.5% for the shared room.

Staten Island balanced with about 50% of full-house and approximately 47% of the shared room ratio.

In this ratio Bronx is closer to Queens, however, we observe an interesting moment in the region. Bronx is a leader in proposing Shared rooms – 5.5% from the total number of listings in the region (for example this relative percentage for Manhattan is 2.2%, for Queens is about 3,5%).



From the correlation map, it is obvious that the room type and location of the renting point have a better correlation with the price compared to other variables.

And finally, using the data, we built a couple of models to predict the price of the rent from the region, neighbourhood and room type, providing a system to predict competitive prices for Airbnb room listings based on locality in NYC (thanks to Multivariate Linear Regression, K-means clustering and CRISP-DM methodology).

Statistical analysis was performed using Python libraries and the data were clearly visualised using charts and graphs in section 5 of this report.

Correlation between different variables was studied and a multivariate linear regression model was created to predict the price of the Airbnb room. Unsupervised machine learning model was built using K-means clustering that could be used as a tool to predict competitive price for new listings in a given locality.

9 Conclusion

For the given dataset, Exploratory Data Analysis was performed and a predictive model was built using Linear Regression and Unsupervised Machine Learning, with Python libraries.

For instance, this model helps to analyse the pricing, understand customers and hosts behaviour, guide in creating marketing ideas, and implement new additional services.

10 References

Kolomatsky, M. (August 14, 2021) What Happened to Airbnb During the Pandemic? *The New York Times*. Available at: https://www.nytimes.com/2021/07/15/realestate/what-happened-to-airbnb-during-the-pandemic.html [Accessed October 26, 2022]

Airbnb second quarter 2022 financial results (August 3, 2022) Available at: https://news.airbnb.com/en-uk/airbnb-second-quarter-2022-financial-results/ [Accessed October 26, 2022]

Dudás, G., Vida, G., Kovalcsik, T. and Boros, L. (2017) A socio-economic analysis of Airbnb in New York City. *Regional Statistics*, *7*(1), pp.135-151.

Lehr, D.D. (2015) An analysis of the changing competitive landscape in the hotel industry regarding Airbnb.

11 Appendix

- Dataset file from https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data
- Code: Used Jupyter notebook for the complete analysis. All the code has been added in sequence from the beginning of the document as a support to the analysis